WIRELESS DISTRIBUTED MONITORING AND CONTROL FOR RECONFIGURABLE PRODUCTION LINES USING DDS BASED MIDDLEWARE

BY

MUHAMMAD NASEER BAJWA

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER ENGINEERING

December 2014

WIRELESS DISTRIBUTED MONITORING AND CONTROL FOR RECONFIGURABLE PRODUCTION LINES USING DDS BASED MIDDLEWARE

by

MUHAMMAD NASEER BAJWA

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

In Partial Fulfillment of the Requirements
for the degree

MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

KING FAHD UNIVERSITY
OF PETROLEUM & MINERALS

Dhahran, Saudi Arabia

DECEMBER 2014

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **MUHAMMAD NASEER BAJWA** under the direction of his thesis adviser and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER ENGINEERING**.

Thesis Committee

7 Dec. 14
Dr. Basem Almadani (Adviser)

Dr. Shuang-Hua Yang (Member)

Dr. Abdul Wahid Al-Saif
(Member) Dec. 7/2014

Dr. Ahmad Almulhem
Department Chairman

Dr. Salam A. Zummo
Dean of Graduate Studies
11/12/14
Date

سُبْحَانَكَ لَا عِلْمَ لَنَآ إِلَّا مَا عَلَّمْتَنَآ ۖ إِنَّكَ أَنتَ ٱلْعَلِيمُ ٱلْحَكِيمُ

All Extolment be to You (O our Lord)! We have no knowledge except that which You have taught us. In truth, it is Thou (and only Thou) Who art perfect in Knowledge and Wisdom.

The Holy Qur'an, 2-32

To All my Mentors
Every bit of me is a little bit of you

ACKNOWLEDGMENTS

الحمد لله رب العالمين والصلاة والسلام على أشرف الأنبياء والمرسلين وشفيع المذنبين سيدنا و مولانا محمد وعلى آله وأصحابه وأزواجه وذريته وأهل بيته أجمعين

It is absolutely inevitable to extend my deepest gratitude and warmest affection to my family and friends who, very graciously and voluntarily, waived their rights on me to let me pursue my studies. I am, and will always be, heavily indebted to them for their continuous support, understanding and ever-needed prayers.

Special credit is due to my thesis advisor Dr. Basem Almadani and committee members Dr. Abdul Wahed Al-Saif and Dr. Shuang-Hua Yang for their constant guidance and motivation right from the onset till the end of this research. Without their honest criticism and painstaking revisions it would not have been possible to produce any quality work.

I am sincerely grateful to KFUPM for granting fully funded scholarship for my Master's degree. The university and specially Computer Engineering Department has been very kind and supportive in arranging necessary equipment and lab setup for the experimentation. Finally, I would also like to thank RTI Inc. for very benevolently providing free license of their middleware product for this project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

THESIS ABSTRACT

NAME: Muhammad Naseer Bajwa

TITLE OF STUDY: Wireless Distributed Monitoring and Control for Reconfigurable Production Lines using DDS based Middleware

MAJOR FIELD: Computer Engineering

DATE OF DEGREE: December 2014

Reconfigurable Manufacturing Systems (RMS) are rapidly becoming choice of production and manufacturing industry due to their quick adaptability to the ever-changing market demands while maintaining the quality and cost of the products. Such systems are usually decentralized in their monitoring and control and consist of heterogeneous components. Therefore, need arises for an interface that can mask the heterogeneity and provide smooth communication among these dissimilar components. Data Distribution Service (DDS) is a data-centric middleware standard based on Real-Time Publish/Subscribe (RTPS) protocol that fulfils the job of such interface in distributed systems. In this work we present

the idea of using DDS-based middleware over commonly used wireless channels like Bluetooth and Industrial WiFi to facilitate data communication in distributed control systems. A simulation model is developed to quantify various performance measures like Latency, Jitter, and Throughput and to examine the suitability of aforementioned wireless channels in distributed monitoring and control environments. The model explores various communication scenarios based upon a practical case study. Obtained results serve as an empirical proof of concept that DDS can ensure reliable and timely data communication in firm real-time distributed control systems using common wireless channels and offer extensive control over various aspects of data transmission through its rich set of QoS policies.

ملخص الرسالة

الاسم الكامل: محمد نصير باجوه

عنوان الرسالة: المراقبة و التحكم الموزع واللاسلكي لاعادة ضبط خطوط الانتاج بواسطة البروتوكول الوسيط المسمي دي دي اس (DDS)

مجال التخصص: هندسة الحاسوب

تاريخ التخريج: صفر ١٤٣٦ هجريه

اصبحت انظمة إعادة ضبط أنظمة التصنيع (RMS) بصورة متزايدة هي الاختيار الافضل الإنتاج والصناعة ويرجع ذلك إلى القدرة على التكيف السريع إلى متطلبات السوق المتغيرة باستمرار مع المحافظة على الجودة والتكلفة للمنتجات. وعادة ما تكون مثل هذه أنظمة اللامركزية في الرصد والمراقبة تتكون من عناصر غير متجانسة. لذلك، تنشأ الحاجة إلى الوسيط الذي يمكن أن يحجب عدم التجانس ويوفر التواصل السلس بين هذه المكونات المتباينة. و و يعتبر الوسيط الامثل لذلك هو الوسيط المعياري المسمى خدمة توزيع البيانات و اختصارا الدي دي اس (DDS) الذي يعتمد في تواصله على البيانات و خاصية النشر و الاشتراك باستخدام بروتوكول (RTPS) المخصص للتطبيقات الحساسة للوقت. في هذا العمل نقوم بتقديم فكرة استخدام البروتوكول الوسيط الدي دي اس مع القنوات اللاسلكية المستخدمة عادة في انظمة التصنيع مثل بلوتوث وواي فاي لتسهيل نقل البيانات في أنظمة التحكم الموزعة. و قد تم تطوير نموذج محاكاة لتحديد مقاييس الأداء المختلفة مثل الوقت المستغرق لوصول البيانات، التقطع في الوصول، و معدل وصول البيانات؛ و تم ايضا دراسة مدى ملاءمة

القنوات اللاسلكية المذكورة في بيئات التحكم والمراقبة الموزعة. و يستكشف ايضا النموذج المقدم سيناريوهات الاتصال المختلفة بالاستناد إلى دراسة حالة عملية خاصة. قد كانت النتائج التي تم الحصول عليها بمثابة دليل عملي على ان مفهوم الدي دي اس يمكنه ضمان تواصل البيانات الموثوق في أنظمة التحكم الموزعة الحساسة للوقت بين الشركات عبر قنوات لاسلكية مشتركة ويمكنه ايضا تقديم سيطرة واسعة على جوانب مختلفة من نقل البيانات من خلال مجموعة غنية من سياسات جودة الخدمة.

درجة الماجستير في العلوم

جامعة الملك فهد للبترول والمعادن

الظهران، المملكة العربية السعودية

صفر ١٤٣٦ هجريه

CHAPTER 1

INTRODUCTION

Manufacturing industry saw a paradigm shift after Henry Ford perfected assembly line in 1913 for his automobile company [1, 2]. The concept was readily adopted in other industries including textile, firearms, sewing machines and locomotives assembly because of its efficiency, machining and improved material handling. These fixed Assembly Lines (FALs) remained most profitable for bulk manufacturing until mid-1990 and are still used wherever mass production of a single part type is required. However, these production lines lack responsiveness to market changes and customer needs – something which has been quite volatile in recent times.

Numerical Control (NC) in automation appeared in early-1950s and later turned into Computer Numerical Control (CNC) machines in 1970s [1]. This technique, called Flexible Manufacturing System (FMS), offers flexibility in producing multiple products on a single system.

These CNC machines, however, could not gain popularity among manufacturers owing to low production rate, low production capacity and high production cost [3].

As globalization and decentralization swept through almost every walk of life after the introduction of internet and powerful, yet cheap, computing devices, industrial automation also benefited from this phenomenal game changer. High data-rate computer networks and powerful embedded systems enabled manufacturing systems to become decentralized and paved the way for Reconfigurable Manufacturing System (RMS). This production technique provides a bridge between the shortcomings of former two techniques, namely Fixed Assembly Lines and Flexible Manufacturing Systems.

Reconfigurable Manufacturing Systems have the capability to quickly respond to launch of new products with scalable production capacities while maintaining product quality and cost [4, 5]. Although the rate of production of these systems is not as high as FALs, yet it is far higher than FMSs. Table 1.1 shows a comparison [6] of aforementioned three techniques and summarizes their pros and cons.

Table 1.1: Comparison of Different Manufacturing Techniques

	FAL	**FMS**	**RMS**
Flexibility	None	High	Medium
Scalability	None	Moderate	High
Production Rate	Very High	Very Low	High
Production Cost	Very Low	Very High	Low

1.1 Challenges in Distributed Control Systems

The Reconfigurable Manufacturing Systems draw their characteristics from Distributed Control Systems (DCSs) which are inherently flexible, agile, adaptable and have no Single-Point-of-Failure. Modern controllers today have become powerful enough to collect data, make decisions and issue commands all on their own instead of sending the data to a centralized control unit. However, this decentralized control has its own challenges that must be addressed at the onset of designing such systems to reap maximum benefits out of this approach.

One of these challenges is that DCSs not only require I/O communication for every controller but also need horizontal (with other controllers on the same hierarchical level) and vertical (with other devices on different hierarchical levels) communication [7]. There is five-level hierarchy in automated production plants as shown in figure 1.1 [8]. Devices communicate with one another on the same level as well as across hierarchy. Network protocols may be different from one level to another. It entails that communicating devices must be able to speak different protocols and have enough bandwidth in order to successfully communicate with other devices.

Another challenge is the heterogeneity [9] of various components

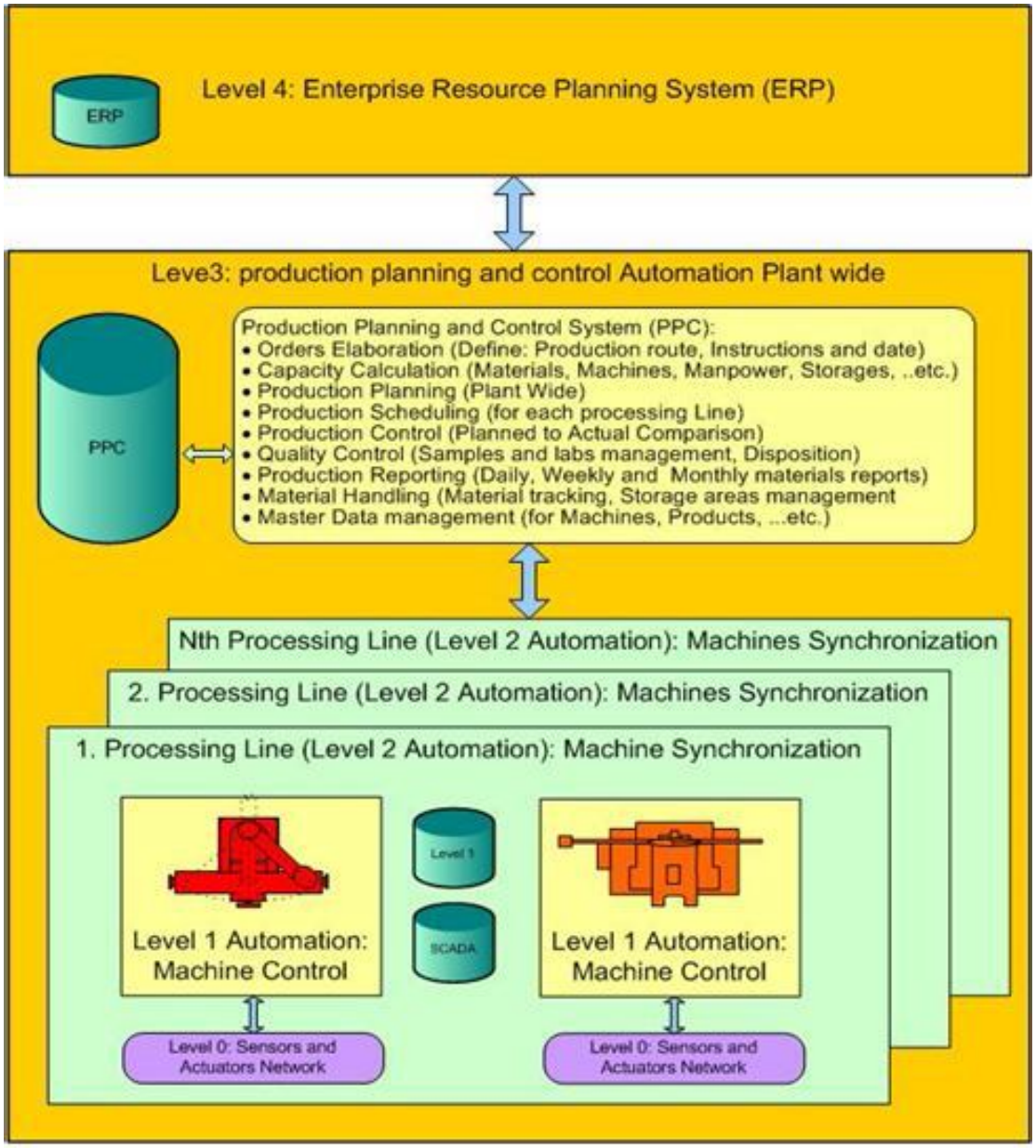

Figure 1.1: Automation Levels in Production Plant [8]

used to constitute the DCS. Figure 1.2 shows a distributed control system architecture used in meat processing industry. The components in this system, normally provided by different vendors, vary in their capabilities, data formats, mapping schemes and I/O interfaces and, therefore, present a rather complex heterogeneous system to deal with.

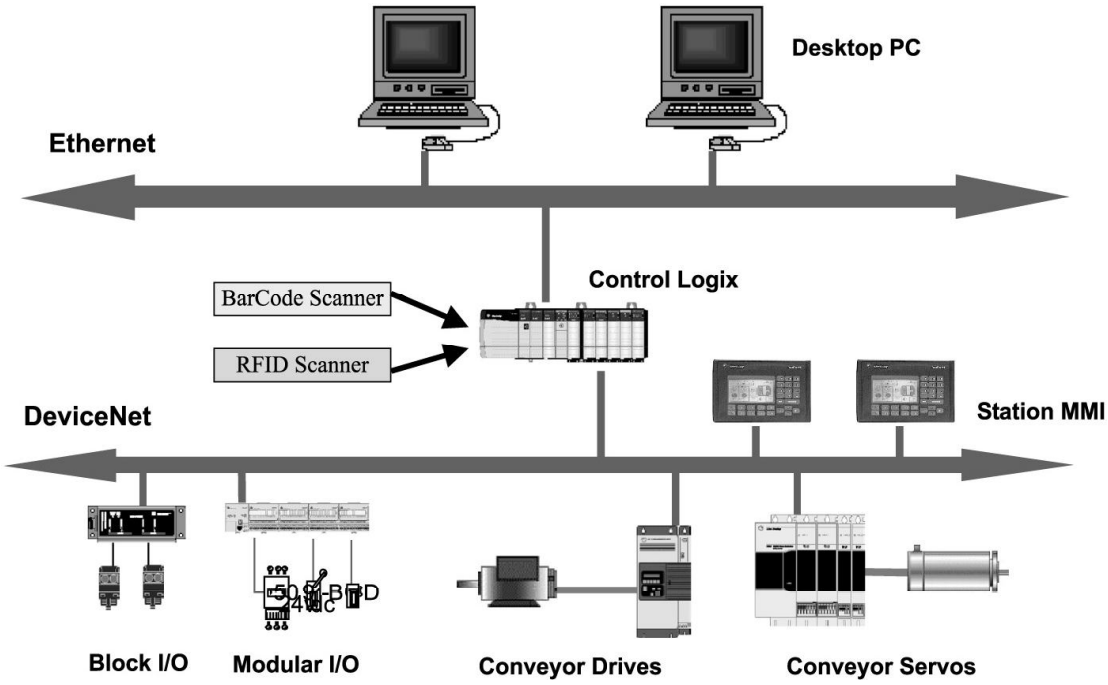

Figure 1.2: Control System Architecture for a Meat Processing Unit [10]

The system depicted in figure 1.2 shows three bottom level hierarchies described in figure 1.1. Also note that the components in this control system are connected through wired media i.e. DeviceNet and Ethernet. If this system were to be designed as RMS, this wired media will pose a great difficulty in physical reconfiguration of system components.

1.2 Middleware Technology

To facilitate the industrial control communication in DCS and mask the heterogeneity amongst the subsystems, various middleware technologies have been proposed over the last couple of decades. These include

Web Services, CORBA (Common Object Request Broker Architecture), Java RMI (Remote Method Invocation) and OPC (OLE for Process Control) etc. These technologies can simplify the design significantly and integrate control devices despite their heterogeneity. However, these solutions lose their value in real-time environments because of their inability to adapt to certain characteristics of real-time process data like periodic messages with data sampled values [7]. Furthermore, they are non-deterministic and don't usually respect strict timelines.

Data Distribution Service (DDS), developed by Object Management Group (OMG), is an open and platform-independent middleware standard that uses Real-Time Publish/Subscribe (RTPS) protocol. DDS-based middleware deploys many-to-many communication model and offers extensive control over a large spectrum of Quality of Services (QoS) policies [9, 11]. Although DDS is a relatively new middleware specification it is gaining substantial attention from researchers as a suitable solution in mission-critical industrial automation applications [12–15]. Figure 1.3 shows the Information Flow in RTPS based middleware. It can be seen from the figure that this is the decoupling of sender and receiver and the use of a distributed Global Data Space that gives DDS-based middleware an edge over other middleware technologies.

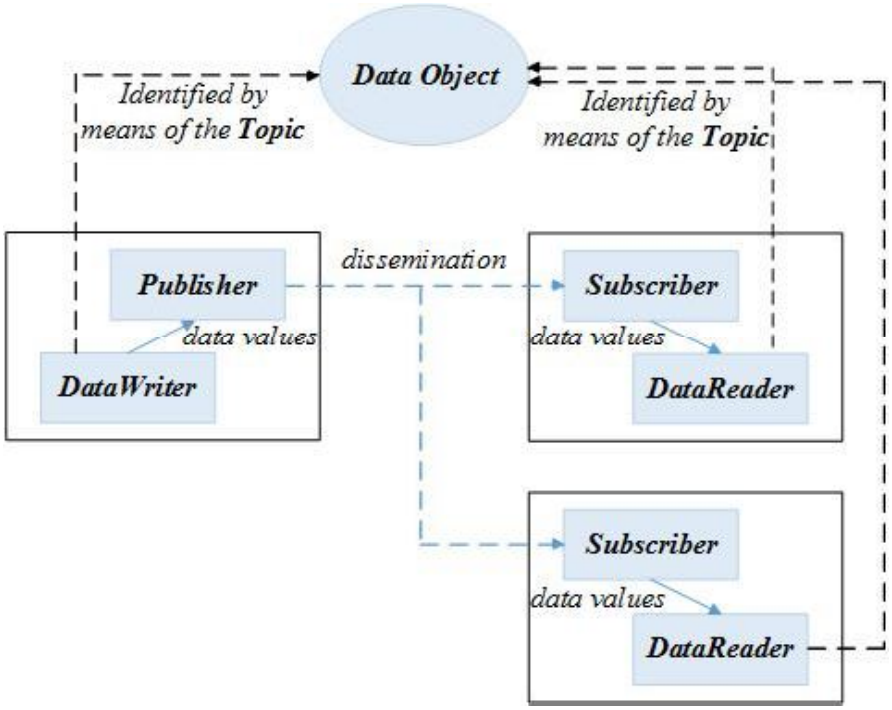

Figure 1.3: Information Flow in RTPS based Middleware

1.3 Wireless Monitoring and Control

Wireless channel seems to be the natural choice as a communication channel in such reconfigurable environments because wired media will severely limit physical reconfiguration of the system components. Let Reconfigurable Production Line (RPL) be defined as a specific type of reconfigurable manufacturing systems spanning a smaller geographical area and able to physically reorient itself to produce a range of products. For such RPLs a short-range limited-bandwidth wireless channel like Bluetooth, Zigbee or WiFi could be a suitable option. In this work Bluetooth and WiFi have been selected as the communication channels and we intend to measure the performance of DDS over these channels

for control and monitoring data.

1.4 Research Objectives

The project necessitates thorough and comprehensive understanding of the DDS based middleware which is at heart of all modern distributed heterogeneous systems. Injecting RTPS data over Bluetooth and WiFi has not been extensively investigated.

The purpose of this work is to empirically provide comparative analysis of different wireless channels against performance measures such as Latency, Jitter and Throughput; and show that DDS standard based middleware can help these wireless technologies meet real-time reliable and efficient data transmission requirements in reconfigurable production lines, despite their low bandwidth constraint, and sustain practical QoS support in majority of applications. The research also aims at proving that DDS based middleware facilitates scalability and masking of heterogeneity in such environment while providing a rich set of QoS policies to control the overall communication. The research has the following objectives to achieve.

1. Performance measurements for control of RPL over Bluetooth using RTPS based middleware

2. Performance measurements for controlling RPL over Industrial

WiFi using RTPS based middleware

3. Comparison of the two technologies against various figures of merit discussed above

4. Provide proof of concept that RTPS based middleware can meet firm real-time requirements in mission-critical RPL environment.

1.5 Research Methodology

The research involves formulating software model of RPL and designing experimental setup to gather results. As the project proposes to compare different wireless channels against some performance measures like Latency, Jitter and Throughput, therefore, testing the data communication over these channels using RTPS based middleware is inevitable to collect actual measurements and compare the wireless technologies.

Experimental setup is developed based on real-time reconfigurable production lines. Software model of control components is developed using C++ API for RTPS based middleware and actual data is transmitted across these components to obtain measurement results of different wireless channels.

1.6 Thesis Breakdown

Having presented the rationale for the research problem and laid down the foundation of the area the problem belongs to in the first chapter, the second chapter will briefly explain the background technologies and fundamental concepts used in this work. Chapter 3 will mention some of the most promising works already done in this field of study to show the big picture and assert the significance and relevance of our work in comparison with existing framework. In chapter 4, our approach to using wireless technologies in RPL environment using DDS based middleware will be discussed in depth. This chapter will also explain the simulation model based upon a practical case study for experimental purpose. Chapter 5 will present experimental setup developed and obtained results. A detailed discussion and analysis of the results will follow to show that DDS based middleware has been able to meet stringent temporal requirements of a firm real-time mission-critical RPL. Finally, chapter 6 will conclude the finding and hint future vistas for further research.

CHAPTER 2

BACKGROUND ON RELATED TECHNOLOGIES

In this chapter we will develop a sound understanding of various technologies and terminologies used throughout the rest of this thesis. Some of the concepts explained in this chapter may be primitive for learned reader but it is always useful to refresh the memory.

2.1 Fundamentals of DDS

Although middleware technology has been in use for last four decades or so [16], first explicitly and then implicitly in various roles and shapes, most of the early models like CORBA, OPC, Java RMI and Web Services, defined in their respective standards in [17–20], use either client-service communication model or message passing communica-

tion paradigm and hence prove somewhat ineffective in real-time environment or are platform-dependent. To make middleware more acceptable in real-time and heterogeneous applications, OMG released DDS [21] as a platform-independent real-time Publish/Subscribe middleware standard capable of implementing a broad set of mechanism to define and manage QoS requirements. Based on these exhaustively elaborative specifications, many implementations are developed that enable various programming language to be combined with commonly used general purpose operating systems. Some open source implementations of DDS include Open DDS and OpenSplice whereas Connext is a proprietary implementation by Real Time Innovation Inc (RTI).

DDS employs two levels interfaces which are;

- DCPS: Data-Centric Publish Subscribe is the lower level of the two interfaces and is responsible for efficient delivery of proper data to the concerned receiver.

- DLRL: Data Local Reconstruction Layer is an optional higher-lever interface which allows integration of the middleware and the application layer.

DCPS ensures predictability and high performance of middleware and efficient use of the system resources. DLRL automatically reconstructs the data based upon the updated values and gives the application an impression as if the data were local. In this way, the middleware

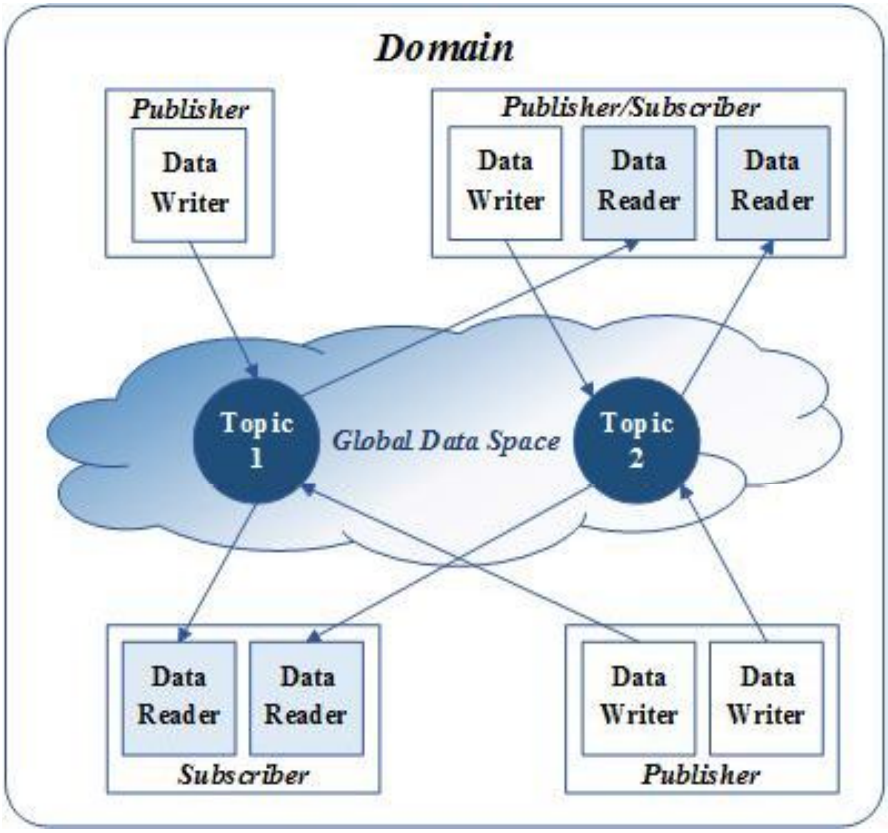

Figure 2.1: DDS Core Entities and their Relationship in DCPS

is enabled to transmit the data to all participating subscribers as well as update a local copy of the information. DDS offers a great deal of flexibility and scalability to the distributed systems by effectively decoupling the publishers and subscribers from each other. Conceptual Outline and the basic constructs used by DCPS in the information flow are shown in Figure 2.1. Brief description of each of them is given below.

- **Topic:** A topic is an information unit that is exchanged across the distributed system. Every Topic is identified by a unique name within a domain and has an exclusive data type.

- **Domain:** A domain is a communication context. It is a distributed concept that links all those applications that can communication with each other. In other words, all the participants belonging to a certain domain can interact only to each other and not to the participants in another domain.

- **Domain Participants:** Participants are the entities involved in data communication in any application. They represent the local membership of the application in a certain domain.

- **Publisher:** A publisher is an object responsible for data distribution. It can publish various types of data using Data Writers. Domain Participants create Publishers to manage a group of Data Writers.

- **Subscriber:** A subscriber is an object responsible for receiving the published data and making it available for the application that the data are supposed to reach. It can receive different types of data using Data Readers. Domain Participants create subscribers to manage a group of Data Readers.

- **Data Writers:** Data Writer is a typed-based object that the middleware uses to communicate the existence and the value of the data object of a certain type. *Typed-based* means that each Data Writer object is associated to only one data type or Topic. If a publisher

wants to publish more types of data it must have dedicated Data Writers for each of the Topics.

- **Data Readers:** Data Readers are typed-based objects that a subscriber must use to accessed subscribed data. Each Data Reader can read only a single Topic and different Data Readers need to be attached to a subscriber in order to receive multiple data types. Therefore, a *subscription* may be defined as an correspondence between a Data Reader and a Subscriber.

Every time an application intends to publish data, it must create a Publisher, or use an existing one, and connect it with a Data Writer that matches the characteristics of the desired publication. Similarly, when an application intends to start receiving updates on a particular Topic it must create a Subscriber, or reuse an existing one, and link it to an appropriate Data Reader.

2.1.1 Mathematical Model of RTPS Paradigm

In this subsection we will briefly discuss the mathematical model of simple publish/subscribe (PS) communication, which is at heart of DDS standard. Zahi *et. al* [22] has elaborated the interaction of various entities in PS model with one another using Set Theory. Let we have three actors participating in PS system: Publisher, Subscriber and Information Repository. Publishers and Subscribers are already

explained earlier. Information Repository is responsible for defining acknowledgements. There are three types of objects: Notification, Subscription and Acknowledgement. Publisher issues notification about its publication, Subscriber defines its requirements for subscription. If the notification about a certain publication matches with any of the subscription request, that publication is delivered to the requesting Subscriber.

Let B be a 6-tuple defined as

$$B = (P, S, I, N, U, A)$$

where

P is a set of Publishers,

S is a set of Subscribers,

I is a set of Information Repositories,

N is a set of Notifications,

U is a set of Subscriptions, and

A is a set of Acknowledgements

B sketches the structure of Publish/Subscribe system and marks the boundaries of system's state space. The three entities present in the system interact with one another by performing certain action. We

define an *event* as an action that changes system's state. Naturally, PS system behaves as discrete event systems. Therefore, we let $E = \{e_1, e_2, e_3, ..., e_i, ...\}$ be a set of, possibly, infinite events that may occur. Every event e_i occurs at a discrete point in time represented by $t(e_i)$. No two events can occur simultaneously i.e. if $t(e_i) = t(e_j)$ then $e_i = e_j$. Hence, we can only have an ordered sequence of events in which e_i always precedes e_j iff $t(e_i) < t(e_j)$ given that $i < j$.

In very primitive model of a PS system following types of events can occur.

Publish A notification is published by Publisher

Notify Subscriber is notified about publication

Subscribe Subscriber activates a subscription

Unsubscribe Subscriber revokes an existing subscription

Acknowledge Information Repository issues acknowledgement

Having laid down the bases for nomenclature, now we can mathematically define a Publish/Subscribe as $PS = (B, E)$ where B describes the structure of the system and E determines its behaviour. PS system behaviour can now be modeled as ordered sequence of events that results in change of system states. System state changes when, as an effect of certain event, any of the individual Publisher, Subscriber

or Information Repository change its state. For example, when a Publisher, P_i, issues a new notification, it changes set of notifications associated with this publisher, $N(P_i)$ triggering a transition in P_i's state. This change in $N(P_i)$ ultimately causes change in $N(P)$ which is super set of $N(P_i)$ and represents set of all notifications issued by all publishers in the system. In the same way the state of a subscriber S_j changes as a result of changes in $N(S_j)$ and $U(S_j)$; and any change in $R(P_i)$ causes change of state of Information Repository I_s.

2.1.2 QoS Support in DDS Specification

OMG has provided an exhaustive set of QoS policies to specifically tailor and govern the behaviour of the communication. These QoS policies can be configured on all DDS entities from as general as Topics and Domain Participants to as specific as Data Writers and Data Readers. DDS provides requirement-specific communication when QoS policy values in sender and receiver side are configured in compatibility to each other because these QoS policies follow Subscriber-requested Publisher-offered pattern. Table 2.1 shows some of the supported QoS policies. In this table Concerns fields specifies which DDS entity is affected by a certain QoS policy. ROC field stands for Request/Offer Compatibility. It signifies whether or not QoS policy values must be compatible on both sender and receiver sides. Changeable field refers

to whether a QoS policy value can be changed dynamically or not.

Table 2.1: Supported QoS Policies

QoS Policy	Description	Concerns	ROC	Changeable
USER_DATA	User data unknown to middleware, but disseminated by built-in topics	DP, DW, DR	No	Yes
TOPIC_DATA	User data unknown to middleware, but disseminated by built-in topics	Topic	No	Yes
GROUP_DATA	User data unknown to middleware, but disseminated by built-in topics	Pub, Sub	No	Yes
DURABILITY	Expresses if the data samples should be kept for late-joining data readers or not	Topic, DW, DR	Yes	No
DURABILITY_SERVICE	Specifies configuration of the durability service as TRANSIENT or PERSISTENT	Topic, DW	No	No
LIFESPAN	The maximum validity duration of a data instance	Topic, DW	N/A	Yes

Continued on next page

QoS Policy	Description	Concerns	ROC	Changeable
HISTORY	Specifies whether to deliver the most recent update or all the intermediate changes	Topic, DR, DW	No	No
DEADLINE	Write or receive a new sample at least once every deadline period	Topic, DR, DW	Yes	Yes
LATENCY_BUDGET	Defines the maximum acceptable delay from the instance the data is written until received by data reader	Topic, DW, DR	Yes	Yes
TRANSPORT_PRIORITY	Determines the priority of the underlying transport	Topic, DW	N/A	Yes
OWNERSHIP	Decides whether and how multiple data writers are allowed to modify the same instance of data	Topic, DW, DR	Yes	No
OWNERSHIP_STRENGTH	Sets the value of the strength to arbitrate which data writer to modify the instance.	DW	N/A	Yes
LIVELINENESS	Defines whether an entity is still active	Topic, DR, DW	Yes	No
PRESENTATION	Determines how to present the change of instance to subscribers	Pub, Sub	Yes	No

Continued on next page

QoS Policy	Description	Concerns	ROC	Changeable
PARTITION	Logical partition of the topics	Pub, Sub	No	Yes
RELIABILITY	Selects between RELIABLE or BEST_EFFORT transmission	Topic, DR, DW	Yes	No
DESTINATION_ ORDER	Specifies the logical order of the changes made to the instance	Topic, DR, DW	Yes	No
RESOURCE_ LIMITS	Determines the resource available to be consumed	Topic, DR, DW	No	No
TIME_BASED_ FILTER	Expresses interest in a particular subset of the data instances	DR	N/A	Yes

In the above table, DP stands for Domain Participant, DW stands Data Writer and DR stands for Data Reader.

2.1.3 Advantages of DDS based Middleware

As a networking middleware, DDS provides a model for sending and receiving messages, events and commands in the network. The key features of DDS based middleware [23, 24] are presented as below.

Data-Centric Approach: As mentioned before, DDS is a data-centric middleware and it employs a fully distributed Global Data Space to store the data. Thus DDS could improve the communication time

and avoid single-point- of-failure.

Connectionless Service: The Real-Time Publish-Subscribe (RTPS) wire protocol used by DDS based middleware is inherently connectionless. Therefore, there is no need to establish point-to-point connection in the network. This feature enables DDS to require relatively low cost and effort to integrate and scale a large system as compared to traditional systems with point-to-point connection.

Automatic discovery: The use of DCPS model in DDS based middleware renders complete decoupling of senders and receivers. It necessitates that these anonymous nodes must be able to *discover* each other and establish communication. DDS provides automatic discovery mechanism which allows subscribers to find and subscriber to any Topic without needing to know the physical existence of its Publisher. In this way any participant can join or leave the network at any time.

Explicit model: One of the most promising feature of DDS is its ability to mask heterogeneity. Communicating applications don't need to know the actual communication mechanism of the DDS because it handles message packing, delivery and unpacking etc. on its own without bothering high level applications.

Interoperability: DDS standard was developed with keeping interoperability in mind. This is achieved via standard DDS APIs, the RTPS wire protocol and the QoS. DDS APIs enable the developers to choose

from a broad range of programming languages and OS platform to design their applications. The RTPS wire protocol specifies the important aspects of DDS, including dynamic discovery and platform independence, etc. while the QoS provides different configurations of QoS policies to satisfy different communication requirements. Over all, these three aspects define all the necessary parts of the standard and ensure the interoperable implementation of DDS.

Rich set of QoS policies: OMG has provided an exhaustive set of QoS policies to manage and control the communication according to any given scenario.

2.2 Basics of Bluethooth

Bluetooth wireless technology is characterized by robustness, low power and low cost [25]. The Bluetooth core system consists of physical layer, baseband and protocol stack. It enables electronic devices to connect and share audio/video and other format of data in a relatively shorter range.

The physical layer of the Bluetooth system uses an RF transceiver, operating in unlicensed ISM band at 2.4 GHz, and employs frequency hopping techniques to avoid interference and fading. RF operation uses a shaped, binary frequency modulation to reduce transceiver complexity. Normally a physical radio channel is shared by many Bluetooth

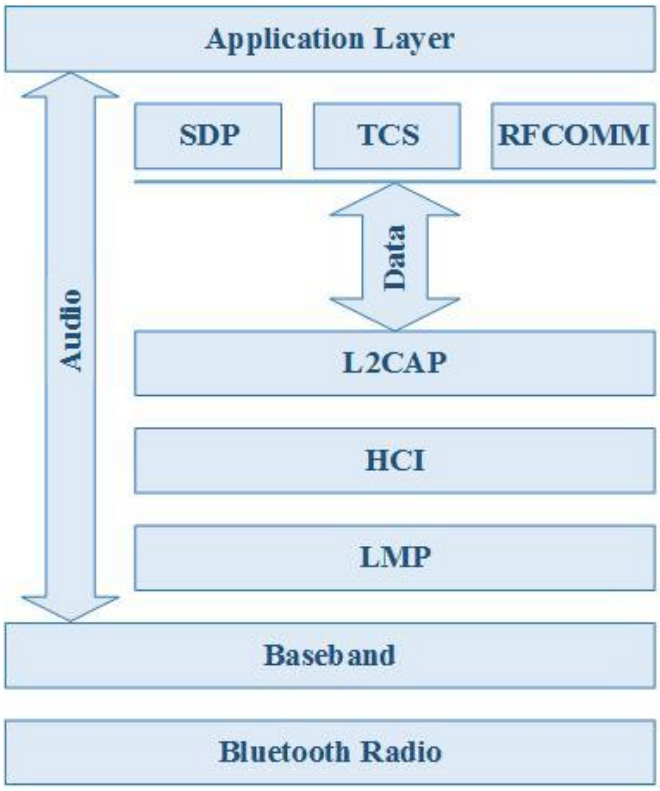

Figure 2.2: Bluetooth Protocol Stack

enabled devices that are synchronized to a common clock and an algorithmically deterministic frequency hopping pattern. The device that provides reference synchronization becomes Master and all other devices are turned into Slaves. Synchronization of a group of devices in this way forms a *piconet*. The physical radio channel is time division multiplexed and during an allocated time slot, the data packets are transmitted using frequency hopping. The Bluetooth allows for Full Duplex transmission by means of Time-Division Duplex (TDD) Scheme.

Within the physical channel, a physical link is created between two devices when they connect to each other. This physical link acts as a transport between the connected devices and multiple logical links can

be established to begin data transfer. Aside from the user data, these logical links also carry Link Manager Protocol (LMP) a control protocol for physical and baseband layer.

Logical Link Control and Adaptation Layer Protocol (L2CAP) resides on top of baseband and is responsible for actual data-exchange. It provides channel-based abstraction to applications and services and performs segmentation and reintegration of application data. The Bluetooth uses Service Discovery Protocol (SDP) to find and communicate with other devices. SDP runs over L2CAP and on even higher level, there are TCP/IP and application protocols.

2.2.1 Bluetooth 3.0+HS

Bluetooth version 3.0+HS (HS stands for High Speed) was adopted by Bluetooth Special Interest Group (SIG) in 2009. It provides data rate as higher as 24 Mbps. This high speed data transfer is made possible using a new feature called AMP (Alternative MAC/PHY). AMP allows Bluetooth 3.0+HS to employ collocated 802.11 link as high speed transport and use its own radio merely for device discovery, link establishment and profile configuration. In other words, when the system is idle Bluetooth's low power connection model is in use and when large amount of data needs to be sent, it can hire services of faster radio link of 802.11. This new feature, however, requires some enhancements in

L2CAP. These include Enhanced Retransmission Model (ERM) for reliable communication and Streaming Model (SM) to implement unreliable communication without flow control.

2.3 Industrial WiFi

WiFi has become a popular choice for connecting industrial devices. It is a well-established, mature technology that offers close integration with existing Ethernet networks. However, the issues that face an industrial system integrator are quite different from the issues encountered in a typical office or commercial application.

Industrial WiFi, like standard WiFi, is based on IEEE 802.11n standard. The implementation of this standard for industrial use aims at resisting various effects of harsh industrial environments, such as high electromagnetic immunity to repel electromagnetic disturbances and galvanic isolation to guard against voltage instability etc. It operates in 2.4 GHz ISM frequency band. Some implementations allow upto four MIMO streams with each stream offering data rate of as high as 150 Mbps. Aggregate data rate is upto 600 Mbps. It has effective range of about 70 meters in indoor environments.

CHAPTER 3

LITERATURE SURVEY

This chapter highlights some of the works involving middleware for real-time distributed environment and discusses the suitability of DDS standard and wireless channels in industrial automation scenarios.

The importance of the middleware technology in distributed systems has been greatly emphasized in [16]. The authors have presented brief and comprehensive history of the middleware and have chronicled the evolution of the technology right since its inception in early 1970s. It was regarded as an inevitable component of any large-scale distributed system. In their words, "trying to build a distributed application without middleware is like trying to write a simple application on a personal computer without the operating system". The authors foresee that the technology will become even more indispensable for distributed heterogeneous system as the systems will tend to grow more complex in the days to come.

Object Management Group's Data Distribution Service is a standard upon which QoS-enabled real-time middleware can be architected. Xiong *et. al* [26] described three possible architectures based on DDS and evaluated the implementation of those platforms to investigate their design tradeoffs. They compared the performance of their implementations with each other and with other Publish/Subscribe middleware. It was concluded, based on the experimentation, that DDS-based implementations perform substantially better than their non-DDS counterparts and are generally well suited for real-time data-critical distributed environment.

Mastouri *et. al* [27] studied the performance of Publish/Subscribe based middleware for real-time distributed control systems. They pointed out that the amount of data and the pattern of transmission call for the Publish/Subscribe architecture which is able to realize many-to-many communication model. The authors examined performance of DCS including DDS in case the communication link is broken and calculated the loss rate while allowing for the caching size.

In [15] Poza *et. al* realized that QoS support usually available in communication layer merely provides simple temporal parameters like message delay or congestion control and, therefore, are insufficient in such scenarios where real-time support, information optimization or component abstraction is desired. They proposed a middleware archi-

tecture, named Frame Sensor Adaptor to Control (FSA-Ctrl), consisting of two layers: a DDS based communication layer and the other Sensor Web Enablement (SWE) based control layer. Both layers sandwich a rich set of QoS policies that empowers control layer to take important decisions about distributed questions like component mobility or information redundancy detection. They implemented their proposed architecture on a home automation problem.

Although DDS is a very powerful and flexible technology, it may prove to be rather complex to fully comprehend – particularly for novice users. This issue was spotted and dealt with by Calvo *et. al* in [13]. The authors floated the idea of a software component encapsulating the functionality of commonly used industrial automation controllers like PLCs, IPCs and Robots which can then be used to create any automation application. The role of DDS in this case can simply be as a communication backbone. The paper shows how to map different traffic patterns using DDS entities taking full advantage of DDS QoS policies.

In [11] a DDS based middleware architecture was proposed by Almadani *et. al* to integrate heterogeneous manufacturing systems such as SCADA, DCS and PLCs. The architecture features improved communication and real-time data delivery across various components and provides a QoS support layer among the communication devices

by adding a small piece of code, termed as *DDS Router*, to interface the OPC protocol on all the integrated components. The DDS Router changes inter-process communication from client/server model to publish/subscribe model, hence effectively reducing data delivery time. In another work [9], Almadani *et. al* conducted performance enhancement of limited-bandwidth wireless industrial control system. They carried out experiments to study various performance measures of control data communication using DDS over Bluetooth. Although the latency over Bluetooth was found to be higher than that over LAN, yet the incentives of wireless communication and greatly reduced packet loss in Bluetooth were thought to make the approach attractive in majority of practical industrial automation scenarios. However, there was no experimental or software model provided in this work. So it cannot be determined how the obtained results will be affected by spatial or topological variations is the physical system.

Realizing that the next generation real-time distributed systems will be highly complex high-performance environments consisting of large number of nodes with heterogeneous characteristics, the authors in [28] proposed an approach to reduce the complexity by using iLand [29] which is an enhanced version of middleware for real-time configuration of service-oriented distributed systems. It has been recognized that future real-time reconfigurable distributed systems are expected to

offer data-intensive capabilities by means of assimilating the processing power of large number of nodes. The systems are projected to gain increased dynamic behaviour as a result of recurrent reconfigurations, for instance. The approach presented in this work involves modelling the reconfiguration of the system as graphs containing all tentative solutions and finding a new valid solution from the complete graph every time the system undergoes reconfiguration. The results show that solution provides phenomenal decrease in the computation time of the reconfiguration process.

The IEC 61499 is an open standard for next generation distributed control and automation systems. Yang and Nolte [24] have built a model of real-time distributed system based on this open architecture. In this work the performances of the RTPS protocol, used by DDS, and that of traditional socket-based communication have been compared. Ethernet has been used as the communication channel. It was discovered through experimentation that, for their defined model, latency, RTT and jitter of socket-based communication is far better than that of DDS communication.

Large scale mobile networks find their applications in a variety of areas like emergency response, logistics, transportation management and environmental monitoring etc. These systems normally require real-time tracking of all the nodes and some means of interaction among

them. The use of DDS based middleware in such large-scale mobile networks has been studied by David *et. al* in [30]. The authors have presented a middleware based on DDS specifications that supports real-time tracking of several thousand mobile vehicles that can communicate with one another and with gateway devices using Unicast, Multicast or Broadcast modes of communication. The underlying communication service employs highly optimized UDP based solution with small footprint. However, their use of LAN as communication medium in mobile networks is arguable. Moreover, Latency results in Broadcast mode are of the order of few seconds which may be tolerable in soft real-time applications but is not acceptable in firm or hard real-time mission-critical applications like battle drones tracking and control.

Another similar work is presented in [31] in which a more refined middleware, termed as Scalable Data Distribution Layer (SDDL), derived from DDS specifications is used for online tracking and monitoring of large scale mobile vehicle fleet spread over vast geographical area. SDDL uses two communication protocols: RTPS protocol for communication with SDDL core network over wired media and Mobile Reliable UDP for wireless communication between core network and mobile nodes. The results confirm that the proposed middleware supports mobile nodes handover, multicast and broadcast communication models in real-time with acceptable round-trip time delays.

DDS based middleware have proven their worth in adaptive real-time video transmission as well. Video monitoring can be of vital importance in industrial control and can stand for human presence and act as remote eye in such situations where it's not environmentally feasible or financially viable for humans to be there. Owing to the real-time nature of video surveillance and non-uniform data-rate, network resource management can become rather crafty. Garcia *et. al* [32] have shown that DDS can provide a potential solution in such cases. They proposed an enhanced middleware framework to support real-time video transmission.

For large scale mobile system, a middleware called Scalable context-Aware middleware for mobiLe EnvironmentS (SALES) is designed by Corradi *et. al* [33]. It's a tree based classified model of nodes for performance and load balancing and calculates communication cost among four types of nodes: Central Node, the Base Node, Coordinator User Node and Simple User Node. SCALES does not take advantage of real-time DDS and totally depend upon UDP. Two main terminologies are used; one is QOC (Quality of Context) and the other is CDDLA (Context data Distribution Level Agreement). QOC is associated with context information distributive service whereas CDSDLA is quality agreement between consumer and producer imposed by the middleware. This SALES architecture, however, lacks the functionality of Fault tolerance,

QoS support and Context Updates by each mobile node.

In [34] a middleware named Solar is proposed for ubiquitous computing. It uses two protocols. TCP for Interplanetary communication and DHT pastry Distributed Hash Table for routing and discovery. It is built on the basis of self-organizing peer-to-peer network and was designed for scalability among the set of communicating nodes. It uses filter and pipe programming model and each filter has group of entry and exit points as well as sources and sinks. Each node in Solar Architecture is assumed as a planet and each planet has a number of Satellite nodes. Increasing the nodes will increase the scalability of this system. Solar has the reliability of TCP only which is not suitable for asynchronous wireless networks. It also lacks the functionality of fault tolerance.

Very little research has been done in implementation of mobile distributed applications through DDS based middleware. Among few of them one is called DDSS [35] middleware. The DDSS architecture supports mobile nodes and provides reliable data delivery. It also supports handover by switching the wireless access points. The mobile nodes in DDSS middleware execute light version of DDS whereas the fixed nodes execute full version of DDS and are responsible for routing and data delivery among the nodes. Due to the design of architecture the mobile nodes have to run in a single domain and stable wireless connectivity is

a must. This architecture also lacks the functionality of fault tolerance.

Revenge (REliable and VErsatile News delivery support for aGEncies) [36] is a DDS based middleware which serves as news dispatching service among the mobile nodes. It efficiently balances the data distribution among the mobile nodes in DDS network. It is based on self-organizing peer-to-peer network and is also fault tolerant. It efficiently detects the crashed nodes and reroutes the paths from any source to any sink. Revenge architecture, however, lacks the functionality of handover. As the mobile nodes also run DDS so it provides DDS QoS as well. In asynchronous environment it supports DDS Publisher/Subscriber property and mobile nodes have tendency to become group Publisher/Subscriber on their own initiative.

Another DDS based middleware is proposed for real time communication between, mobile nodes using proxy approach [37]. In this architecture all the mobile nodes run light version of the DDS client whereas fixed nodes run the full version of DDS Client. A PROXY DDS Client is used for managing, coordinating and forwarding all the data to mobile nodes. This architecture provides both reliable and unreliable data delivery. Due to Firewall/NAT restrictions all the mobile nodes have to run in single domain and continuous connectivity is needed.

A middleware based on Publish/Subscribe model to be used with RFID is introduced in [38]. Since different applications require dif-

ferent data, middleware has to adapt to all applications. When more applications are in need, it is necessary to adjust middleware to satisfy them, which, however, costs lots of time and efforts. By use of Publish/Subscribe mechanism, applications can subscribe to events from the RFID reader dynamically. In Publish/Subscribe model filtering of messages can be topic based or content based. In Topic-based filtering the subscriber will receive messages published to a specific Topic and to the Topic to which they subscribed. All the subscribers will receive same messages for a particular Topic. In content-based filtering subscriber will receive messages if the content of messages match the content constraints set by the subscribers. Four components are proposed for Publish/Subscribe model for this RFID middleware: (1) List of publications (maintained by the message manager), (2) List of subscriptions (maintained by the message manager), (3) Message manager (A controller responds to the requests of reader manager for maintenance as well as client queries for List of publication. It also maintains the list of subscriptions), and (4) API (set of routines and protocols, interface that clients implement subscriptions and unsubscriptions).

The suitability of short-range limited-bandwidth communication channels, like Zigbee and Bluetooth, in industrial applications has been studied by Nick Baker in [39]. The author discussed various merits and demerits of both technologies on different grounds. It is es-

tablished that because of relatively greater data-rate and faster active slave channel access, Bluetooth is better than Zigbee for machine-to-machine communication and for ad hoc connectivity between the fixed equipment and mobile devices.

Advantages of using Bluetooth and Wifi over DDS in traffic monitoring application has been recognized in [40]. In this recent work, DDS has been used in a vast distributed system of mobile vehicles to automatically locate and monitor vehicle movement in the holy city of Makkah during the times of annual pilgrimage when million of Muslims gather in the city to perform the ritual. Enormous amount of traffic at various choke points cause a lot of trouble for pedestrians and vehicles alike. Locating a certain pilgrim or group of pilgrims in this huge crowd is also challenging task. The authors have proposed a solution in which each pilgrim travelling to holy sites via bus is identified by an RFID tag which contains his/her personal particulars. Each bus has mechanism to read the RFID tags of its passengers and transmit the data using DDS over WiFi to DDS cloud at various checkpoints along the way to the holy sites. The results show that DDS over Bluetooth and WiFi can provide timely data transmission with minimum packet loss and can handle thousands of mobile or stationary nodes at a given instance.

CHAPTER 4

DISTRIBUTED CONTROL OVER WIRELESS MEDIA

In this chapter we present the idea of using DDS over Bluetooth and Industrial WiFi in RPL environment. Study of the related work, as presented in the previous chapter, has revealed that DDS over wireless LAN and Bluetooth has not been examined extensively. There is a need for proof of concept that these wireless technologies can meet stringent time requirements of firm real-time distributed systems like Reconfigurable Production Lines. We start by explaining the rationale for using wireless technology in RMS applications and then develop a software simulation model based upon a simple RMS system. We will conclude the chapter with brief description of the performance measures used in this work.

4.1 DDS over Wireless Channel

Ethernet is widely used in distributed industrial control as the communication channel due to its high-bandwidth and minimal packet loss. To avoid unnecessary wiring and make the system more tidy, WiFi can be used as a wireless substitute for Ethernet; however, it is prone to packet drop with the increase in traffic because it relies upon an access point to transmit the data between communicating nodes. Bluetooth, on the other hand, uses mesh topology that eliminates the need of any such device and offers better reliability in term of data delivery. Though it has comparatively narrower bandwidth, but can still work pretty fine in the given area of application knowing that data rate requirements in industrial sensing and control applications are often low to intermediate [39].

Running DDS over Industrial WiFi is straightforward as most of the DDS-based middleware implementations use UDP/IP transport, by default. This UDP/IP transport integrates smoothly with WiFi network protocol. Figure 4.1 illustrates the layered architecture of DDS over Bluetooth. In the transport layer of DDS, UDP/IP provides the middleware with fine-grained control over data transmission and allows it to decide whether the transmission should be reliable or best effort, depending upon the application requirement and underlying network type.

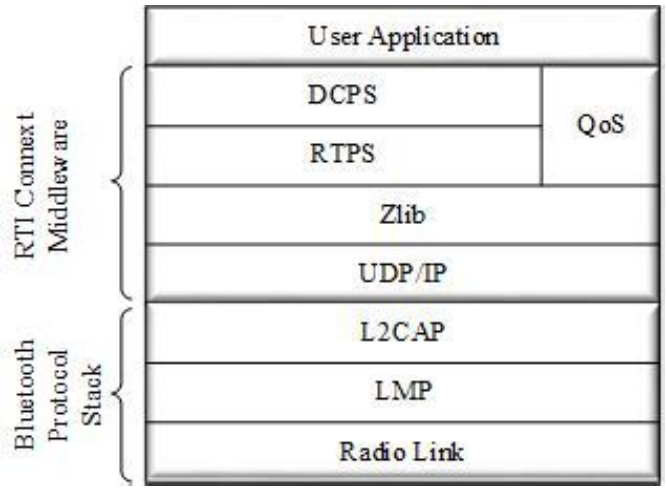

Figure 4.1: Layered Architecture of DDS over Bluetooth

Above UDP/IP layer is Zlib which is a software library that performs data compression. It's an example where RTI DDS developers can implement their very own data compression algorithms to suit their application specific needs. RTPS wire protocol uses a packet header size of 56 bytes that includes timestamps and sub-message headers [21]. This metadata, or transmission overhead, can further be reduced by the governing application – though this area has not been greatly explored yet, especially for low bandwidth channels.

The focus of this work is to empirically show that DDS-based middleware can meet real-time reliable and efficient data transmission requirements in RMS environment over abovementioned wireless channels and sustain practical QoS support in majority of applications. In the remaining of this chapter, we present a test case setting corrosponding to distributed real-time control where judicious selection of QoS policies can lead to smooth, reliable and on-time data transmission

amongst various components of the system.

4.2 A Simple Test Case Setup

To evaluate the performance of DDS-based middleware in RMS, consider the automated manufacturing system shown in Figure 4.2. The system is adapted from [41] and consists of three machines represented by M1, M2 and M3. It has five sensors (S1,S2, S3, Si, So) and four actuators (A1, A2, A3, Ai) for blocking. The manufacturing system is constructed using PLCs. Two types of pallets, Type 1 and Type 2 are input to the system randomly. The route which these pallets may take is decided by a three-position stop, Ai, whose selection depends upon the routing information provided by S1 and S3 sensors.

All three machines publish a set of values pertaining to their operation. These values may include state of the machine and various physical quantities like motor speed, pressure and temperature etc. Machines can be instructed to physically reposition themselves using configuration commands sent by the control panel (not shown in the diagram). Besides issuing configuration commands, control panel also monitors machines data by subscribing to their respective Topics.

Sensors 1, 2 and 3 look for the availability of their respective machines. If a machine is currently operating on some pallet then these sensors publish FALSE Boolean value to indicate that the machine is

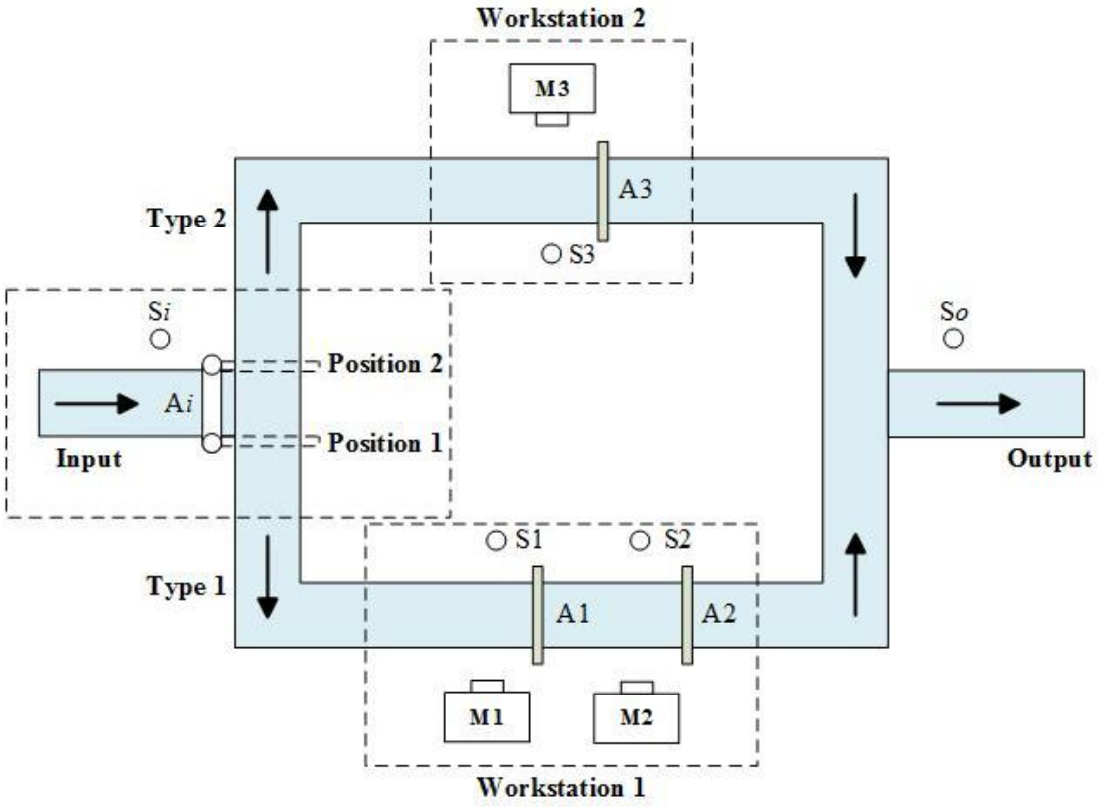

Figure 4.2: An Automated Reconfigurable Manufacturing System

currently busy and no more input may be dispatched on this route. Otherwise they transmit TRUE as soon as the machine is ready for the next input. Input sensor (Si) keeps an eye on input availability to the system. Output Sensor (So) monitors whether a finished product by either workstation has left the system. It helps avoid product collision over the relay.

Actuators 1, 2 and 3 are stop points and remain close as long as corresponding machines are working. They are open to let the finished product pass and new input get into the workstation. Dispatcher Ai is responsible for forwarding the input (if available) to either of the two

routes or hold until one of them is available.

The motors, actuators and sensors can physically relay across the workbench (shown in grey strip) enabling the whole system to transform into a single-, double-, or triple-workstation system thus facilitating configurability of the system in both physical and functional sense. As it is assumed that the whole setup covers a relatively smaller space (up to few tens of meters), therefore, use of limited-bandwidth wireless channel seems proper in this scenario.

4.2.1 Software Model for a Simple RPL

The DCPS model corresponding to the scenario depicted in Figure 4.2 is shown in Figure 4.3. As can be seen from the figure, the model incorporates one-to-many, many-to-one and many-to-many communication requirements. Each rectangle and square represents a domain participant (we are assuming that all the participants are in a single domain). These DDS entities correspond to various types of hardware devices like sensors, actuators and motors as shown in Figure 4.2.

4.3 Performance Measures

Before moving on to the experimentation, it's better to first discuss performance measures that will determine whether Bluetooth can fulfil real-time data communication requirements of the system.

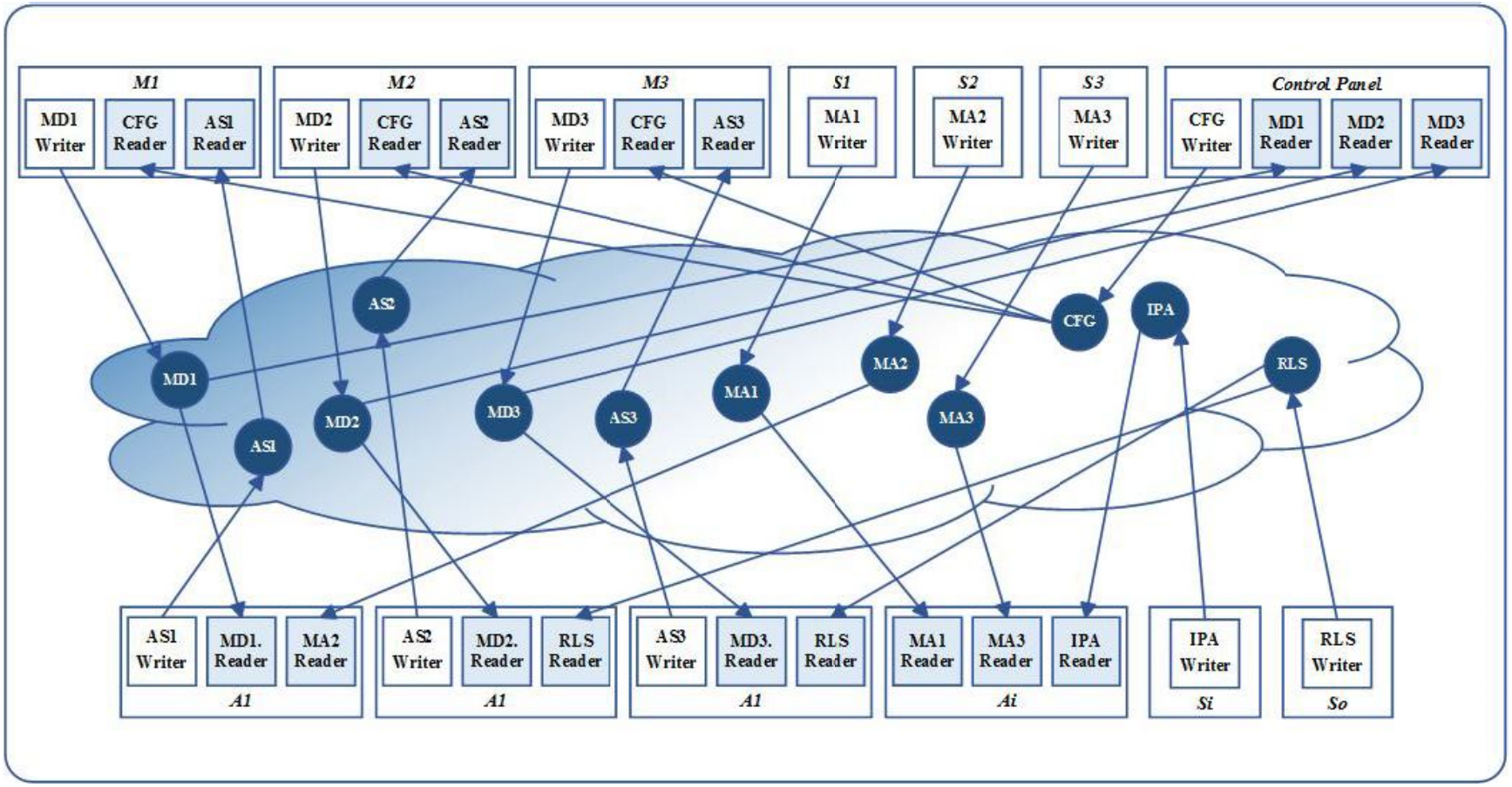

Figure 4.3: DCPS Model Corresponding to the System Shown in Figure 4.2; MDx = Machine Data x, CFG = Configuration Command, MAx = Machine Availability x, ASx = Actuator Status x, RLS = Relay Status, IPA = Input Availability, MDx. = Machine Data x.Status (status member of data structure MDx)

- **Average Latency:** Latency is the time a data packet takes to reach the receiver side. It includes propagation delay plus queuing delay at the receiver side. Average Latency can be calculated by dividing Round Trip Time (RTT) by 2 as given in Equation 4.1.

$$AverageLatency = RTT/2 \tag{4.1}$$

In firm real-time systems, deadlines are relatively relaxed as compared to hard real-time systems. Although a packet arriving after the deadline may not be of any value, however, occasional longer delays or even packet loss does not cause the system to fail [42, 43].

- **Jitter:** Jitter is the variation in latency. Smaller values of jitter mean that the data will, most of the time, experience almost the same amount of delay during its voyage from sender to receiver. Mathematically it can be represented as given in Equation 4.2.

$$Jitter = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{4.2}$$

Here N is the total number of samples and $\bar{x}$ is the mean value of delay.

- **Throughput:** Throughput denotes average rate of successful data transmission over a channel. This data-rate takes not only pay-

load into consideration but also any packet overhead. We used Equation 4.3 to calculate the throughput.

$$Throughput = \frac{PacketSize \times No.of Packets}{TotalTime} \tag{4.3}$$

Bluetooth offers several frame formats with varying header and payload sizes. The most common format has 126 bits of metadata and up to 2744 bits of payload [44]. However for a single time slot of 625 μs (there are 1600 frequency hops per second in Bluetooth) the frame carries only 240 bits as payload while frame metadata remains the same.

CHAPTER 5

EXPERIMENTATION AND RESULTS

For testing and simulation the ***rtiddsgen*** utility provided by RTI Connext 5.0.0 is used to generate C++ code. Visual Studio 2010 is used to build the code and Wireshark 1.2.3 is employed for traffic monitoring over wireless channels. The QoS used in experimentation are given in Table 5.1.

Table 5.1: QoS Used in Experimentation

QoS Policy	**Value**	
	Publisher	**Subscriber**
DURABILITY	TRANSIENT	TRANSIENT
LATENCY_BUDGET	40 ms	40 ms
LIVELINESS	AUTOMATIC	AUTOMATIC
max_lease_duration	120 sec	120 sec
RELIABILITY	BEST_EFFORT	BEST_EFFORT
HISTORY	KEEP_ALL	KEEP_LAST
RESOURCE_LIMITS *max_samples_per_instance*	LENGTH_UNLIMITED	1

5.1 QoS Policies used in Experimentation

Below we present a short description of the QoS used for the experimentation and their interdependence on one another.

- **DURABILITY:** Durability QoS decides if data should outlive their writing time, that is to say whether or not data samples should be archived in middleware service after they are written. A VOLATILE type does not care to save any sent data samples on behalf of Data Writers, however, TRANSIENT type maintains record of sent updates in memory and the data is not tied to the lifecycle of Data Writer. It means that data will still be available even if the corresponding Data Writer goes offline. These achieved samples may be delivered to late-joining subscribers who want to know what they missed.

- **LATENCY_BUDGET:** This QoS policy describes maximum acceptable delay between sending and receiving of the data. This is not something carved in the stone but rather just a guideline to the service. If an updates fails to meet this acceptable delay, the service will not raise any flags or discard the packet.

- **LIVELINESS:** It indicates the mechanism by which the middleware knows if any participating entity is active or has gone offline. Every Data Writer periodically signals its liveliness to all

the Data Readers. The signalling period must not exceed *liveliness_lease_duration* otherwise Data Reader assumes that the Data Writer is no more alive.

- **RELIABILITY:** The RELIABILITY QoS policy specifies the level of reliability that a subscriber can offer or a publisher can request. It has two values, RELIABLE and BEST_EFFORT. In RELIABLE mode, the Data Reader must acknowledge the receiving of each and every packet that arrives. Data Writer does not discard any data value that has been transmitted but not yet acknowledged. This approach has slightly negative effect on the latency because the receiving entity must check the integrity and order of the received packet before acknowledging. It also consumes some of the channel bandwidth for acknowledgements. On the other hand, if a packet drop, every once in a while, does not greatly affect the system and the application cares little about the order of the data received, it's better to use BEST_EFFORT mode which does not require sending or receiving acknowledgements.

- **HISTORY:** This QoS defines the behaviour of middleware service in case the value of the data changes before it is successfully transferred to the receiver. On sender side, it control the number of samples that will be kept with Data Writer on behalf of Data Reader. On Receiver side, it indicates the number of sample main-

tained by Data Reader until the subscriber application reads the data.

- **RESOURCE_LIMIT:** It indicates how much resources the middleware may consume to comply with the QoS requirements. Configuration of this QoS must be in accordance with other QoS settings. For example, if RELIABILITY is set to RELIABLE, then Data Writers need some memory space to store the data samples that has been sent but not yet acknowledged. If we set *max_samples_per_instance* to, say, 1 then Data Writer will not be able to cache enough unacknowledged packet to implement RELIABLE communication. Therefore, to implement RELIABILITY QoS successfully, enough resources must be allocated using RESOURCE_LIMIT.

5.2 Experiments for RTT and Jitter

Our experimentation measures Round Trip Time (RTT) of packets. Realizing the fact that one-way latency is not necessarily always half of RTT, we record only two-way RTT in the following tables. It can be construed, however, with absolute certainty that average one-way latency can never be greater than RTT. We used 1024 Bytes payload (which is the maximum payload size in the given scenario) for experimentation. Following, we briefly explain testing scenarios for One-to-Many and Many-to-Many setup separately.

5.2.1 RTT and Jitter in One-to-Many Setup

First, one publisher and multiple subscribers scenario is examined and latency and jitter are measured. In each run 10,000 to 50,000 packets are sent; the tests are repeated upto 10 times and average is taken to make the results more precise. Table 5.2 shows the results obtained.

Table 5.2: RTT and Jitter for One-to-Many Scenario

Setup	Min (m sec)		Max (m sec)		Average (m sec)		Jitter (m sec)	
	BT	WiFi	BT	WiFi	BT	WiFi	BT	WiFi
1 - 1	15.88	2.56	100.60	127.02	18.22	4.82	9.20	4.70
1 - 2	22.00	2.76	142.16	114.58	25.24	4.94	10.92	3.92
1 - 4	34.62	3.54	128.16	224.88	38.84	8.00	13.42	9.36
1 - 6	47.08	4.12	145.00	908.28	52.32	8.46	15.34	10.32
1 - 8	59.58	4.44	199.12	1414.8	66.90	11.90	17.68	15.26

Based upon the collected data, Jitter is calculated using Equation 4.2. We can see that both RTT and Jitter increase linearly as the number of subscribers grow. Due to higher bandwidth, WiFi RTT is significantly lower than Bluetooth RTT despite the fact that WiFi packets, from one node, must go to an access point before being routed to the destination node. However, the values of average Latency (RTT/2) for Bluetooth, even for the worst case, are well within acceptable range for firm real-time requirements (around 40 msec [45]).

Figures 5.1 and 5.2 show the graphs of RTT and jitter, respectively, corresponding to Table 5.2. We also observe that while WiFi Jitter is far smaller than Bluetooth Jitter for fewer subscribers, the gap between the

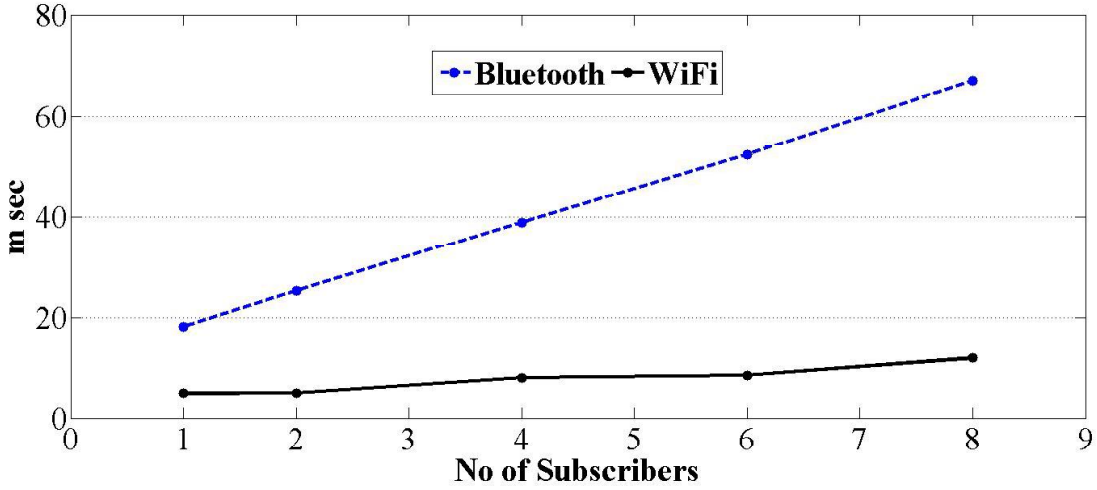

Figure 5.1: Average RTT against Various Number of Subscribers

two tends to shrink as more and more subscribers (and consequently network traffic) joins in and packet drop increases. This is because of the fact that WiFi performance at a certain point in time depends greatly on the amount of traffic at that given instance.

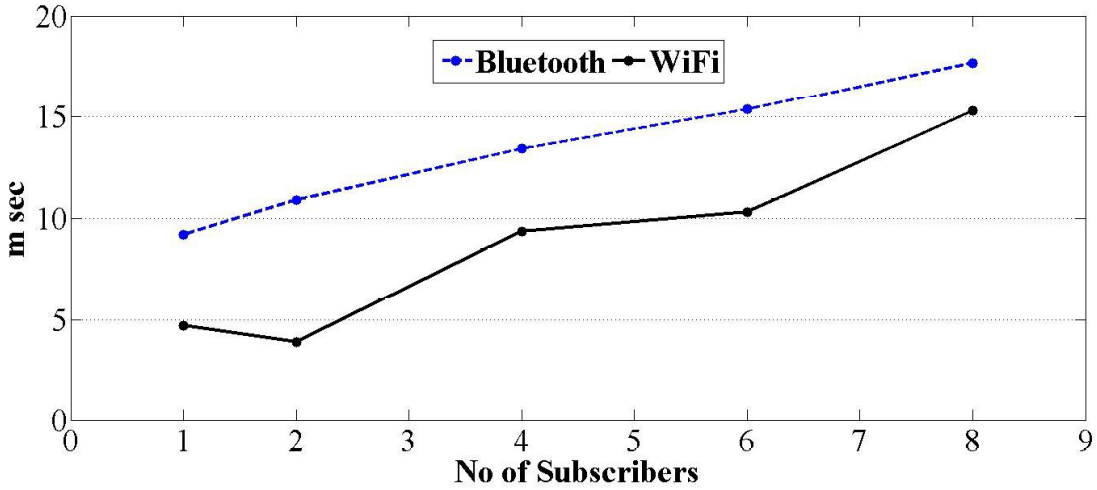

Figure 5.2: Jitter against Various Number of Subscribers

5.2.2 RTT and Jitter in Many-to-Many Setup

RTT and jitter are measured for many-to-many communication scenario as well. Tests are run to examine the effect of multiple publishers and multiple subscribers on the RTT and jitter. As expected, both performance measures have higher values when multiple participants try to transmit the data over a single channel. These results are tabulated in Table 5.3 below.

Table 5.3: RTT and Jitter for Many-to-Many Model

Setup	Min (m sec)		Max (m sec)		Average (m sec)		Jitter (m sec)	
	BT	**WiFi**	**BT**	**WiFi**	**BT**	**WiFi**	**BT**	**WiFi**
2 - 2	17.24	2.84	335.80	263.86	36.68	5.38	17.88	5.90
2 - 4	22.20	3.28	421.16	474.48	38.62	6.86	15.08	6.72
4 - 4	31.88	3.10	502.04	212.96	47.42	6.80	24.62	6.78
4 - 8	44.30	4.86	518.54	583.06	74.10	12.64	19.76	8.56

Figure 5.3 shows the results in graphical format. Here again we can see that DDS ensures small enough Latency and Jitter over both communication channels to accomodate firm real-time requirements of most RPLs. However, WiFi precedes Bluetooth in terms of better Latency and Jitter in every scenario owing to its higher bandwidth.

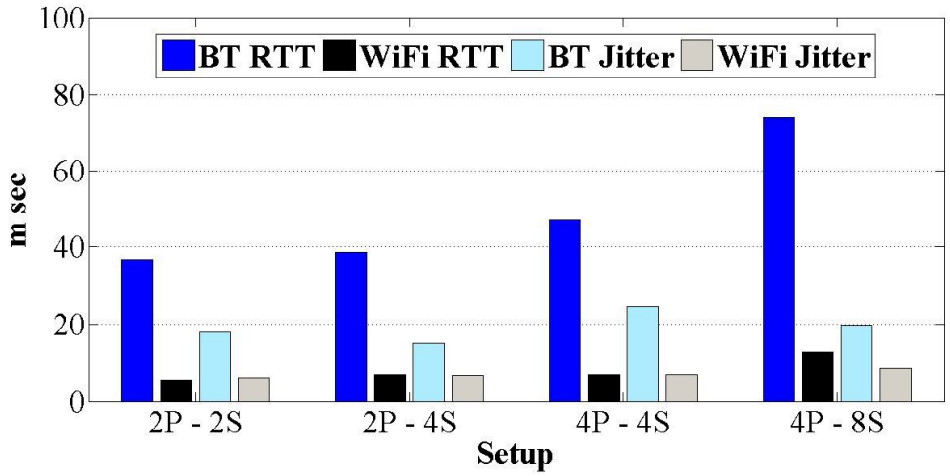

Figure 5.3: Average RTT and Jitter for Various Many-to-Many Scenarios

5.3 Experiments for Throughput

Throughput depends upon the size of the data packets and the frequency of the transmission. Most of the data packets used in the use case are less than 150 bytes. Only configuration commands may extend up to 1 KB. We used this maximum packet size to measure the throughput of both channels. This time, 100,000 to 160,000 samples of data packets are sent from the publisher side and received on the subscriber side in each iteration. Total time for this communication is noted and throughput is calculated using Equation 4.3. The experiments are conducted at least 10 times to get more precise results.

5.3.1 Throughput in One-to-Many Setup

The results collected for one Publisher and many Subscribers are shown in the table 5.4. We notice from Table 5.4 and Figure 5.4 that the

throughput for the given packet size is not quite affected by the number of participants currently active. There is very small and random difference in the throughput against various number of subscribers. Bluetooth provides extremely high throughput for 24 Mbps channel. Though the throughput of WiFi is significantly low yet it provides better latency values for the low data-rate communication.

Table 5.4: Throughput for One-to-Many Model

Setup	**Total Packets (x1000)**		**Total Time (sec)**		**Throughput (Mbps)**	
	BT	**WiFi**	**BT**	**WiFi**	**BT**	**WiFi**
1 - 1	3137	4585	1107	5515	23.22	6.81
1 - 2	3091	2794	1125	3288	22.50	6.96
1 - 4	3156	3330	1110	3805	23.28	7.17
1 - 6	3560	3780	1235	3985	23.61	7.77
1 - 8	3786	5040	1311	5505	23.66	7.50

In these experiments, the maximum number of subscribers are limited to 8. This is for two reasons; firstly, our case study does not required more than 8 number of subscribers for any control data and secondly, the trend in the performance measures can be easily deduced with these many participants. It is anticipated that with the increase in the number of Subscribers latency, jitter and throughput will follow the same course as obtained in these experiments.

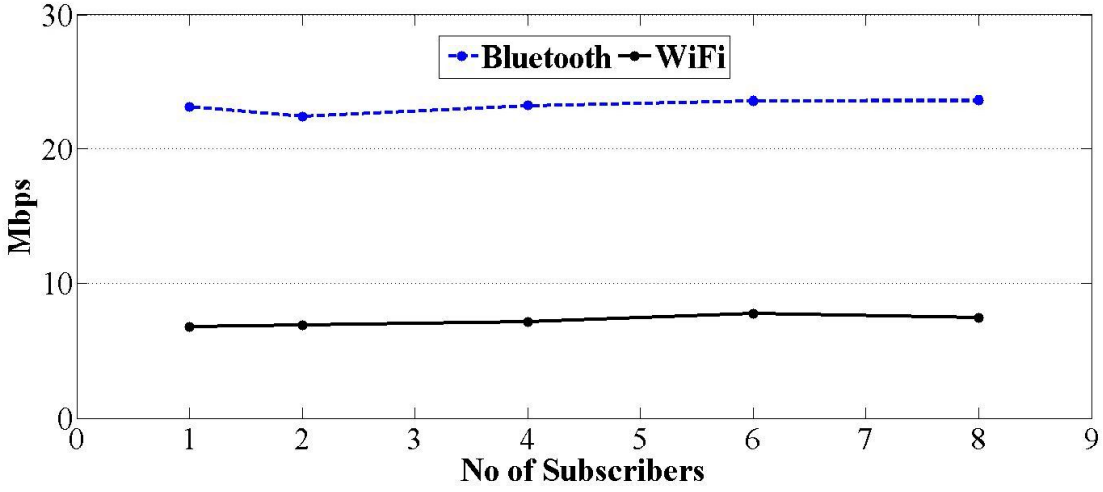

Figure 5.4: Throughput Graph for Single Publisher and Multiple Subscribers

5.3.2 Throughput in Many-to-Many Setup

Like many-to-many latency and jitter experiments, we also conducted tests to measure average throughput over Bluetooth and Industrial WiFi. Various configurations of Publishers and Subscribers are examined. Here again Bluetooth 3.0 HS surpasses WiFi in term of higher average throughput.

Table 5.5: Throughput for Many-to-Many Model

Setup	Total Packets (x1000)		Total Time (sec)		Throughput (Mbps)	
	BT	WiFi	BT	WiFi	BT	WiFi
2 - 2	3368	3560	1238	3706	22.28	7.87
2 - 4	4082	4032	1399	4360	23.90	7.67
4 - 4	4178	4032	1445	2767	23.69	12.37
4 - 8	4208	4032	1445	2730	23.86	12.59

Table 5.5 summarized the results obtained for both channels. We can observe from figure 5.5 that throughput is not significantly affected

by the number of participants in many-to-many scenarios as well.

It should be noted, however, that tests on WiFi require a very controlled environment to avoid any unnecessary traffic and ensure that only DDS applications are using the channel. Bluetooth, on the other hand, does not require such consideration because its point-to-point connection is not affected by overall network traffic..

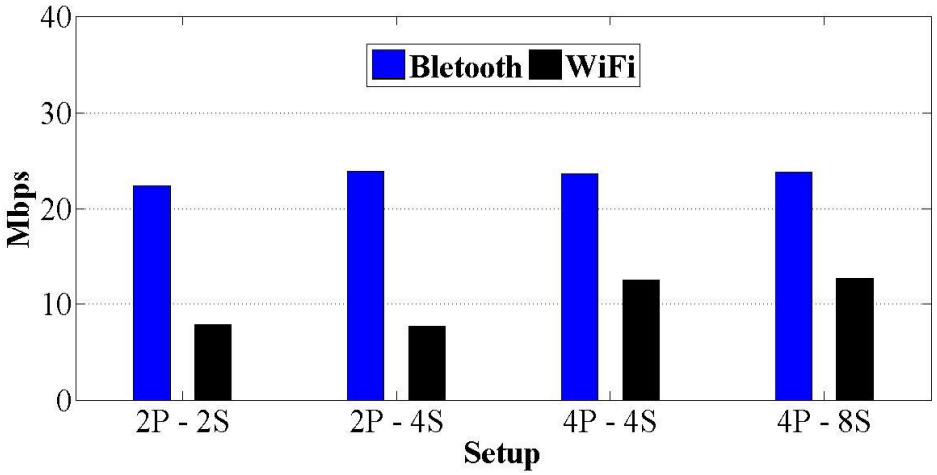

Figure 5.5: Throughput Graph for Multiple Publishers and Multiple Subscribers

CHAPTER 6

CONCLUSION AND FUTURE DIRECTIONS

Most of the distributed industrial control systems involve exchange of simple control parameters and system states, and therefore, require relatively low data-rates. Reconfigurable Production Lines, a form of DCS, usually span over a relatively smaller area and DDS based middleware can mediate among heterogeneous components of such RPL systems by offering a data-centric communication paradigm for abstracting their peculiar data representations. Both WiFi and Bluetooth can be potential candidates for wireless channel among various communicating devices in RPL owning to their low-cost, simplicity, energy efficiency, security and reliable data delivery. RTPS wire protocol of DDS can smoothly interface with both these protocol stacks.

The results show that DDS over the wireless channels like Bluetooth

and Industrial WiFi fulfil real-time data communication requirements of most of the limited-bandwidth small-area control systems. They offers high throughput for small data packets and low latency acceptable in firm real-time systems. These performance measures in conjunction with security and reliability of these media make them a safe and reliable choice for most RPL applications.

For future research work, using DDS over Zigbee in DCS environment can be an interesting area to investigate. Injecting RTPS data over rather low-bandwidth Zigbee network can be challenging. Research in this area can present proof of concept of Zigbee suitability in mission-critical real-time applications using DDS-based middleware. It is, however, expected that latency may increase and throughput significantly decrease given narrower bandwidth of the channel.

REFERENCES

[1] Y. Koren and M. Shpitalni, "Design of Reconfigurable Manufacturing Systems," *Journal of Manufacturing Systems*, vol. 29, no. 4, pp. 130–141, 2010.

[2] "Ford's assembly line starts rolling," http://www.history.com/this-day-in-history/fords-assembly-line-starts-rolling, [Online; accessed 07-September-2014].

[3] A. I. Dashchenko, *Reconfigurable Manufacturing Systems and Transformable Factories*. Springer, 2007.

[4] Y. Koren, *The Global Manufacturing Revolution: Product-Process-Business Integration and Reconfigurable Systems*. John Wiley & Sons, 2010, vol. 80.

[5] M. F. Zaeh, *Enabling Manufacturing Competitiveness and Economic Sustainability*. Springer, 2013.

[6] S. M. Nazir, "Reconfigurable Manufacturing Systems: Key to World-Class Manufacturer Status for Indian Organisations, An

overview," *Journal of Research in Commerce and Management*, vol. 1, no. 3, pp. 87–93, 2012.

[7] I. Calvo, F. Pérez, I. Etxeberria, and G. Morán, "Control Communications with DDS using IEC61499 Service Interface Function Blocks," in *15th Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2010, pp. 1–4.

[8] B. Almadani, "Data Management Systems 530011," Institute for Automation, University of Leoben, 2007, [Lecture 4 Presentation].

[9] B. Al-Madani, A. Al-Roubaiey, and M. F. Al-Hammouri, "Performance Enhancement of Limited-Bandwidth Industrial Control Systems," *Advanced Materials Research*, vol. 739, pp. 608–615, 2013.

[10] A. Mousavi, M. Sarhadi, A. Lenk, and S. Fawcett, "Tracking and Traceability in the Meat Processing Industry: A Solution," *British Food Journal*, vol. 104, no. 1, pp. 7–19, 2002.

[11] B. Almadani, A. Al-Roubaiey, and R. Ahmed, "Manufacturing Systems Integration using Real Time QoS-Aware Middleware," *Advanced Materials Research*, vol. 711, pp. 629–635, 2013.

[12] I. Calvo, F. Pérez, O. G. de Albeniz, and I. Etxeberria-Agiriano, "Towards a OMG DDS Communication Backbone for Factory Automa-

tion Applications," in *16th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2011, pp. 1–4.

[13] I. Calvo, F. Pérez, I. Etxeberria-Agiriano, and O. Garcia de Albeniz, "Designing High Performance Factory Automation Applications on Top of DDS," *International Journal of Advanced Robotic Systems*, vol. 10, 2013.

[14] I. Etxeberria-Agiriano, I. Calvo, F. Pérez, and G. de Albeniz, "Mapping Different Communication Traffic over DDS in Industrial Environments," in *6th Iberian Conference on Information Systems and Technologies (CISTI)*. IEEE, 2011, pp. 1–6.

[15] J. L. Poza, J. L. Posadas, and J. E. Simó, "QoS-based Middleware Architecture for Distributed Control Systems," in *International Symposium on Distributed Computing and Artificial Intelligence (DCAI)*. Springer, 2009, pp. 587–595.

[16] J. Al-Jaroodi and N. Mohamed, "Middleware is STILL Everywhere!!!" *Concurrency and Computation: Practice and Experience*, vol. 24, no. 16, pp. 1919–1926, 2012.

[17] Object Management Group, "CORBA Specifications, version 3.3," http://www.omg.org/spec/CORBA/3.3/, Nov. 2012, [Online; accessed 04-November-2013].

[18] OPC Foundation, http://www.opcfoundation.org/, [Online; accessed 04-November-2013].

[19] Oracle Corporation, "Java Platform Standard Edition 8 Documentation," http://docs.oracle.com/javase/8/docs/index.html, [Online; accessed 07-November-2014].

[20] The World Wide Web Consortium, "Web Service Architecture," http://www.w3.org/TR/ws-arch/, Feb. 2004, [Online; accessed 04-November-2013].

[21] Object Management Group, "Data Distribution Service for Real-time Systems, version 1.2," 2007.

[22] L. Zhai, L. Sun, and Y. Liu, "Modeling and Evaluation of High-Performance Publish-Subscribe System," in *International Symposium on Computational Intelligence and Design (ISCID)*, vol. 1. IEEE, 2008, pp. 457–460.

[23] J. Yang, "Data Distribution Service for Industrial Automation," Master's thesis, School of Innovation, Design and Engineering, Mlardalen University, Sweden, 2012.

[24] J. Yang, K. Sandstrom, T. Nolte, and M. Behnam, "Data Distribution Service for industrial automation," in *17th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2012, pp. 1–8.

[25] Bluetooth Special Interest Group, "Bluetooth Specifications Version 4.1," 2013.

[26] M. Xiong, J. Parsons, J. Edmondson, H. Nguyen, and D. C. Schmidt, "Evaluating the Performance of Publish/Subscribe Platforms for Information Management in Distributed Real-Time and Embedded Systems," 2011.

[27] M. A. Mastouri and S. Hasnaoui, "Performance of a Publish/Subscribe Middleware for the Real-Time Distributed Control Systems," 2007.

[28] M. García-Valls, P. Uriol-Resuela, F. Ibáñez-Vázquez, and P. Basanta-Val, "Low Complexity Reconfiguration for Real-Time Sata-Intensive Service-Oriented Applications," *Future Generation Computer Systems*, vol. 37, pp. 191–200, 2014.

[29] M. Garcia Valls, I. R. López, and L. F. Villar, "iLAND: An Enhanced Middleware for Real-Time Reconfiguration of Service Oriented Distributed Real-Time Systems," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 228–236, 2013.

[30] L. David, R. Vasconcelos, L. Alves, R. André, G. Baptista, and M. Endler, "A Large-Scale Communication Middleware for Fleet Tracking and Management," in *Salao de Ferramentas Simpo-*

sio Brasileiro de Redes de Computadores e Sistemas Distribuidos (SBRC), 2012, pp. 964–971.

[31] L. David, R. Vasconcelos, L. Alves, R. André, and M. Endler, "A DDS-based Middleware for Scalable Tracking, Communication and Collaboration of Mobile Nodes," *Journal of Internet Services and Applications*, vol. 4, no. 1, pp. 1–15, 2013.

[32] M. García-Valls, P. Basanta-Val, and I. Estévez-Ayres, "Adaptive real-time video transmission over DDS," in *8th International Conference on Industrial Informatics (INDIN)*. IEEE, 2010, pp. 130–135.

[33] A. Corradi, M. Fanelli, and L. Foschini, "Adaptive Context Data Distribution with Guaranteed Quality for Mobile Environments," in *5th International Symposium onWireless Pervasive Computing (ISWPC)*. IEEE, 2010, pp. 373–380.

[34] G. Chen, M. Li, and D. Kotz, "Data-Centric Middleware for Context-Aware Pervasive Computing," *Pervasive and mobile computing*, vol. 4, no. 2, pp. 216–253, 2008.

[35] A. Herms, M. Schulze, J. Kaiser, and E. Nett, "Exploiting publish/subscribe communication in wireless mesh networks for industrial scenarios," in *13th International Conference on Emerging*

Technologies and Factory Automation (ETFA). IEEE, 2008, pp. 648–655.

[36] A. Corradi, L. Foschini, and L. Nardelli, "A DDS-Compliant Infrastructure for Fault-Tolerant and Scalable Data Dissemination," in *Symposium on Computers and Communications (ISCC)*. IEEE, 2010, pp. 489–495.

[37] K.-J. Kwon, C.-B. Park, and H. Choi, "A Proxy-Based Approach for Mobility Support in the DDS System," in *6th International Conference on Industrial Informatics (INDIN)*. IEEE, 2008, pp. 1200–1205.

[38] J. Yu and S. Lai, "Message Publish/Subscribe System of RFID Middleware," in *International Conference on New Trends in Information and Service Science (NISS)*. IEEE, 2009, pp. 288–292.

[39] N. Baker, "ZigBee and Bluetooth Strengths and Weaknesses for Industrial Applications," *Computing & Control Engineering Journal*, vol. 16, no. 2, pp. 20–25, 2005.

[40] B. Almadani, S. Khan, T. R. Sheltami, E. M. Shakshuki, M. Musaddiq, and B. Saeed, "Automatic Vehicle Location and Monitoring System based on Data Distribution Service," *Procedia Computer Science*, vol. 37, pp. 127–134, 2014.

[41] M. Nourelfath, D. Ait-kadi, and W. I. Soro, "Availability Model-

ing and Optimization of Reconfigurable Manufacturing Systems," *Journal of Quality in Maintenance Engineering*, vol. 9, no. 3, pp. 284–302, 2003.

[42] T. T. H. Le, L. Palopoli, R. Passerone, Y. Ramadian, and A. Cimatti, "Parametric Analysis of Distributed Firm Real-Time Systems: A Case Study," in *15th Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2010, pp. 1–8.

[43] T. T. H. Le, L. Palopoli, R. Passerone, and Y. Ramadian, "Timed-Automata based Schedulability Analysis for Distributed Firm Real-Time Systems: A Case Study," *International Journal on Software Tools for Technology Transfer*, vol. 15, no. 3, pp. 211–228, 2013.

[44] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks 5th Edition*. Prentice Hall, 2012.

[45] B. Almadani, "Data Management Systems 530011," Institute for Automation, University of Leoben, 2012, [Lecture 7 Presentation].

Muhammad Naseer Bajwa

Vitae

Education

2005–2009 **B.Sc in Computer Engineering**, *COMSATS Institute of Information Technology*, Lahore, Pakistan, *Aggregate Percentage – 83%.*
Graduated with Distinction

Bachelor's Thesis

Title *Active Noise Control*

Supervisor Dr. Sarwar Ehsan

Description This thesis was about effectively detecting, analyzing, and neutralizing the acoustic noise from environment in real-time. The system was modelled in Matlab Simulink and implemented on C6713 DSP Starter Kit.

Experience

2009–2012 **Lab Engineer**, *COMSATS Institute of Information Technology*, Lahore, Pakistan.

- Developed lab experiments in Control Systems, Computer Organization and Design, Advanced Digital Logic Design and Digital Signal Processing courses
- Worked in MultiRate Communication Networks (MRCN) research group on adaptive impulsive noise cancellation in Power Line Communication

Achievements

- Stood 2^{nd} among 41 students in undergraduate program and received Silver Medal
- Earned university's merit based scholarship equal to full tuition fee for undergraduate program
- Won fully funded scholarship for Master's degree

Personal Information

DoB August 06, 1985

Nationality Pakistani

Language Urdu (Native), Punjabi (Native), English (Proficient)

Qadria Market, Wandala Road, Shahdara – Lahore, Pakistan
(+92) 322 477 8910 • ☎ *(+92) 423 793 1408*
✉ *naseeralibajwa@hotmail.com*